Claims:

1. A method for executing a software application in a plurality of computing nodes having node resources, wherein said nodes include a first node and a second node that intercommunicate over a network, and said nodes being operative to execute a virtual machine that runs under a guest operating system, comprising the steps of:

running at least a first virtual machine implementer and a second virtual machine implementer on said first node and said second node, respectively; and

sharing said virtual machine between said first virtual machine implementer and said second virtual machine implementer.

2. The method according to claim 1, further comprising the step of running said software application over said guest operating system, so that commands invoked by said software application are monitored or emulated by said first virtual machine implementer and said second virtual machine implementer on said first node and said second node, while said node resources of said first node and said second node are shared by communication over said network.

3. The method according to claim 1, wherein at least one of said first virtual machine implementer and said second virtual machine implementer is a virtual machine monitor.

4. The method according to claim 1, wherein at least one of said first virtual machine implementer and said second virtual machine implementer is an emulator.

5. The method according to claim 1, wherein at least said first node comprises a first virtual node comprising a first physical CPU of said first node and a second virtual node comprising a second physical CPU of said first node.

6. The method according to claim 1, wherein said virtual machine comprises a first virtual machine and a second virtual machine, wherein said first virtual machine and said second virtual machine have a plurality of virtual CPU's that are virtualized by said first virtual machine implementer based on a first physical CPU and said second virtual machine implementer based on a second physical CPU, respectively.

7. The method according to claim 6, and a first virtual node comprises said first physical CPU and said second physical CPU.

8. The method according to claim 7, wherein said first virtual machine implementer virtualizes at least one of said virtual CPU's of said first virtual machine based on said first physical CPU and virtualizes at least one of said virtual CPU's in said second virtual machine based on said second physical CPU.

9. The method according to claim 1, further comprising the steps of:

providing a management system for said first virtual machine implementer and said second virtual machine implementer to control said first node and said second node, respectively, wherein said management system comprises a wrapper for receiving calls to a device driver from said first virtual machine implementer, said wrapper invoking said device driver

according to a requirement of said first virtual machine implementer.

10. The method according to claim 9, further comprising the step of providing a virtual PCI controller for said management system to control a physical PCI controller in one of said nodes.

11. The method according to claim 9, further comprising the step of providing a virtual DMA controller for said management system to control a physical DMA controller in one of said nodes.

12. The method according to claim 11, further comprising the steps of:

providing a virtual PCI controller to control a physical PCI controller in one of said nodes; and

during a bootup phase of operation scanning a device list with said virtual PCI controller to identify devices having on-board DMA controllers.

13. The method according to claim 1, further comprising the steps of:

with said virtual machine implementer maintaining mirrors of a memory used by said guest operating system in each of said nodes;

write-invalidating at least a portion of a page of said memory in one of said nodes; and

transferring a valid copy of said portion of said page to said one node from another of said nodes via said network.

14. A computer software product, comprising a computer-readable medium in which computer program instructions are stored, which instructions, when read by a computer, cause the computer to perform a method for executing a software application in a plurality of computing nodes having node resources, wherein said nodes include a first node and a second node that intercommunicate over a network, and said nodes being operative to execute a virtual machine that runs under a guest operating system, comprising the steps of:

running at least a first virtual machine implementer and a second virtual machine implementer on said first node and said second node, respectively; and

sharing said virtual machine between said first virtual machine implementer and said second virtual machine implementer.

15. The computer software product according to claim 14, wherein at least one of said first virtual machine implementer and said second virtual machine implementer is a virtual machine monitor.

16. The computer software product according to claim 14, wherein at least one of said first virtual machine implementer and said second virtual machine implementer is an emulator.

17. The computer software product according to claim 14, wherein at least said first node comprises a first virtual node comprising a first physical CPU of said first node and a second virtual node comprising a second physical CPU of said first node.

18. The computer software product according to claim 17, wherein said virtual machine comprises a first virtual machine and a second virtual machine, wherein said first virtual machine and said second virtual machine have a plurality of virtual CPU's that are virtualized by said first virtual machine implementer based on said first physical CPU and said second virtual machine implementer based on said second physical CPU, respectively.

19. The computer software product according to claim 18, wherein said plurality of virtual CPU's that are virtualized by said first virtual machine implementer based on said first physical CPU and said second virtual machine implementer based on said second physical CPU, respectively.

20. The computer software product according to claim 18, wherein said first virtual node comprises said first physical CPU and said second physical CPU.

21. The computer software product according to claim 20, wherein said first virtual machine implementer virtualizes at least one of said virtual CPU's of said first virtual machine based on said first physical CPU and virtualizes at least one of said virtual CPU's in said second virtual machine based on said second physical CPU.

22. The computer software product according to claim 14, wherein said computer is further instructed to perform the step of running said software application over said guest operating system, so that commands invoked by said software application are received by said first virtual machine implementer and said second virtual machine implementer on said first node and said

second node, while said node resources of said first node and said second node are shared by communication over said network.

23. The computer software product according to claim 14, further comprising the steps of:

providing a management system for said first virtual machine implementer and said second virtual machine implementer to control said first node and said second node, respectively, wherein said management system comprises a wrapper for receiving calls to a device driver from said first virtual machine implementer and said second virtual machine implementer, said wrapper invoking said device driver according to a requirement of said first virtual machine implementer and said second virtual machine implementer.

24. The computer software product according to claim 23, further comprising the step of providing a virtual PCI controller for said management system to control a physical PCI controller in one of said nodes.

25. The computer software product according to claim 23, wherein said computer is further instructed to perform the step of providing a virtual DMA controller for said management system to control a physical DMA controller in one of said nodes.

26. The computer software product according to claim 25, wherein said computer is further instructed to perform the steps of:

providing a virtual PCI controller to control a physical PCI controller in one of said nodes; and

during a bootup phase of operation scanning a device list with said virtual PCI controller to identify devices having on-board DMA controllers.

27. The computer software product according to claim 14, wherein said computer is further instructed to perform the steps of:

with said virtual machine implementer maintaining mirrors of a memory used by said guest operating system in each of said nodes;

write-invalidating at least a portion of a page of said memory in one of said nodes; and

transferring a valid copy of said portion of said page to said one node from another of said nodes via said network.

28. A computer system for executing a software application, comprising:

a plurality of computing nodes, having node resources, said plurality of computing nodes comprising at least a first node and a second node;

a network connected to said first node and said second node providing intercommunication therebetween;

said first node and said second node being operative to execute a first virtual machine implementer and a second virtual machine implementer respectively, wherein a virtual machine is implemented concurrently by at least said first virtual machine implementer and said second virtual machine implementer; and

said nodes being operative to execute a guest operating system over said virtual machine,wherein said software application executes over said guest operating system, so that commands invoked by said software application are received by

said first virtual machine implementer and said second virtual machine implementer on said first node and said second node, while said node resources of said first node and said second node are shared by communication over said network.

29. The computer system according to claim 28, wherein said software application comprises a first software application and a second software application, said guest operating system comprises a first guest operating system and a second guest operating system, and said virtual machine comprises a first virtual machine and a second virtual machine, wherein said first software application and said first guest operating system are associated with said first virtual machine, and said second software application and said second guest operating system are associated with said second virtual machine.

30. The computer system according to claim 29, wherein one of said nodes has a first physical CPU and a second physical CPU, and said first virtual machine implementer virtualizes a first virtual CPU in said first virtual machine based on said first physical CPU and virtualizes a second virtual CPU in said second virtual machine based on said second physical CPU.

31. The computer system according to claim 28, wherein at least said first node comprises a first virtual node and a second virtual node.

32. The computer system according to claim 31, wherein said first node comprises a first processor and a second processor, a first I/O device and a second I/O device, wherein said first I/O device is assigned to said first processor, and said second I/O device is assigned to said second processor.

33. The computer system according to claim 28, further comprising a minimal operating system executing in each of said nodes to invoke said first virtual machine implementer and said second virtual machine implementer so that said first virtual machine implementer and said second virtual machine implementer control said nodes.

34. The computer system according to claim 28, further comprising a management system for said first virtual machine implementer and said second virtual machine implementer to control said first node and said second node, respectively, wherein said management system comprises a wrapper for receiving calls to a device driver from said first virtual machine implementer and said second virtual machine implementer, said wrapper invoking said device driver according to a requirement of said first virtual machine implementer and said second virtual machine implementer.

35. The computer system according to claim 34, further comprising a virtual PCI controller for said management system to control a physical PCI controller in one of said nodes.

36. The computer system according to claim 34, further comprising a virtual DMA controller for said management system to control a physical DMA controller in one of said nodes.

37. The computer system according to claim 28, further comprising a memory management system executing in at least one of said nodes that maintains mirrors of a memory used by said guest operating system in each of said nodes, wherein said memory management system write-invalidates at least a portion

of a page of said memory in one of said nodes; and transfers a valid copy of said portion of said page to said one node from another of said nodes via said network.